WAVEFORM SIMULATION

Elizabeth Worcester (BNL)

BNL DUNE Meeting

May 10, 2017

Context

- Work so far has been motivated by SBND ADC decision
- Interesting to evaluate impact of P1 ADC imperfections and calibration schemes as this ASIC will be used in protoDUNE
- Growing interest in a dual-gain solution for DUNE FD

- This presentation is ~same as presented in SBND ADC ASIC committee meeting last Thursday
  - Added a few comments
  - Added some updates at end

Simulation Basics

- Goal: at the single waveform level, evaluate impact of measured P1 ADC performance
- Inputs:
  - Simulated parallel MIP pulses from SBND (Michelle)
  - Analyzed LN test bench data (D. Adams)
    - mV to ADC distribution
    - Good/bad chip characterization
    - Bad code identification (>1 mV RMS)
    - Calibration data for each channel
    - Currently using ~10 chips with a few channels per chip for my simple simulation, easily expandable.
    - David working on fully integrated simulation: https://indico.fnal.gov/conferenceDisplay.py?confId=14463
- Algorithm:
  - Read in pulse in mV (single pulse for now)
  - Sample from ADC distribution
  - Analyze resulting waveform in various ways…
  - Use calibration to convert back to mV for comparison to true waveform

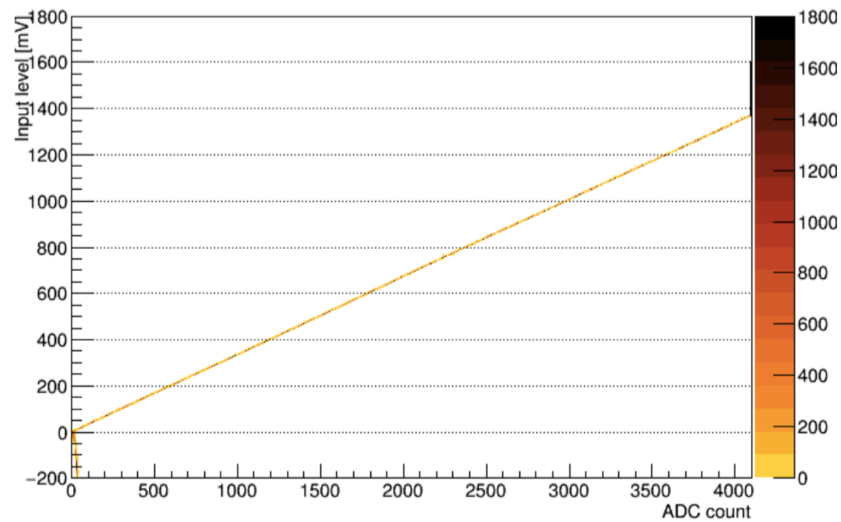
Inputs

Read in simulated pulse (SBND

Notes about this waveform:

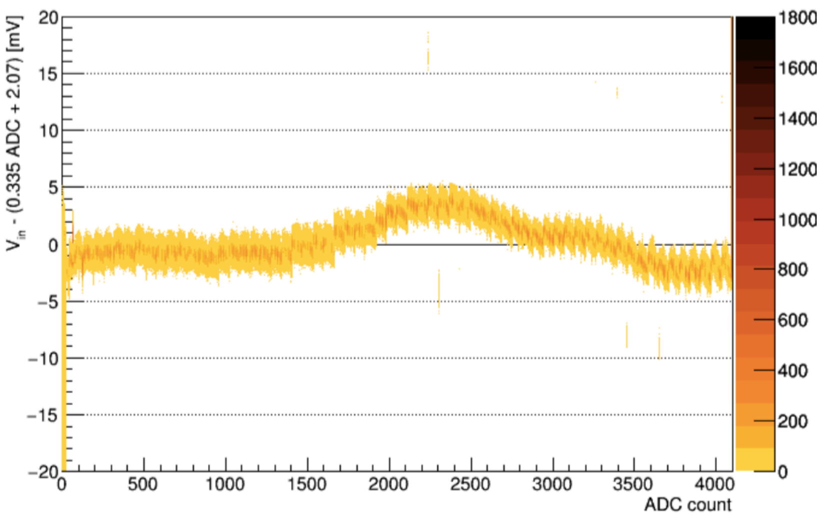
- Noise level is too low – Michelle measures RMS = 0.61 mV, should be 0.86 mV (for 385 ENC, extrapolated from uboone noise to SBND wire length) – Michelle re-running with higher noise levels
- Only parallel MIPS – Michelle re-running with some higher track angles
- White noise applied after shaping time simulation (this is standard in LArSoft) so a bit unphysical – acquired stand-alone noise simulation from Mike Mooney and Adam Lister (uboone) for future studies

Randomly select a chip and channel. For each tick in waveform, sample ADC value from this distribution (measured on bench for each channel). Convert back to mV using linear gain measurement (fit to this distribution).

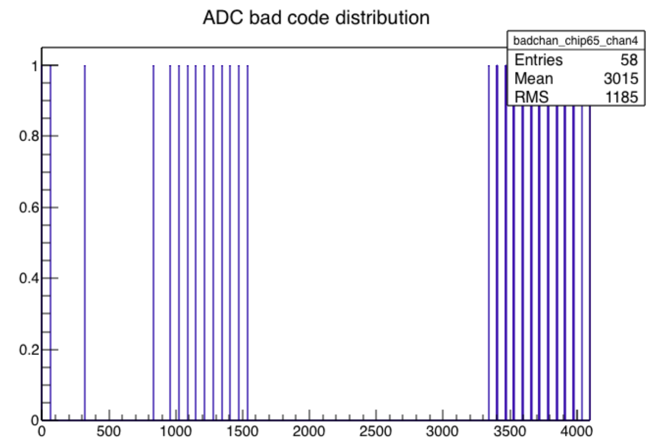


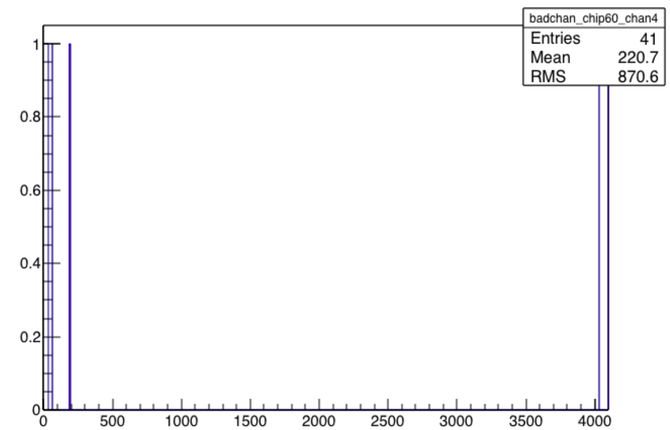
Inputs (cont)

May apply correction based on mean of measured residuals after linear calibration (4096 bins for each channel).



May remove/replace data for "bad codes" (identified for each channel)

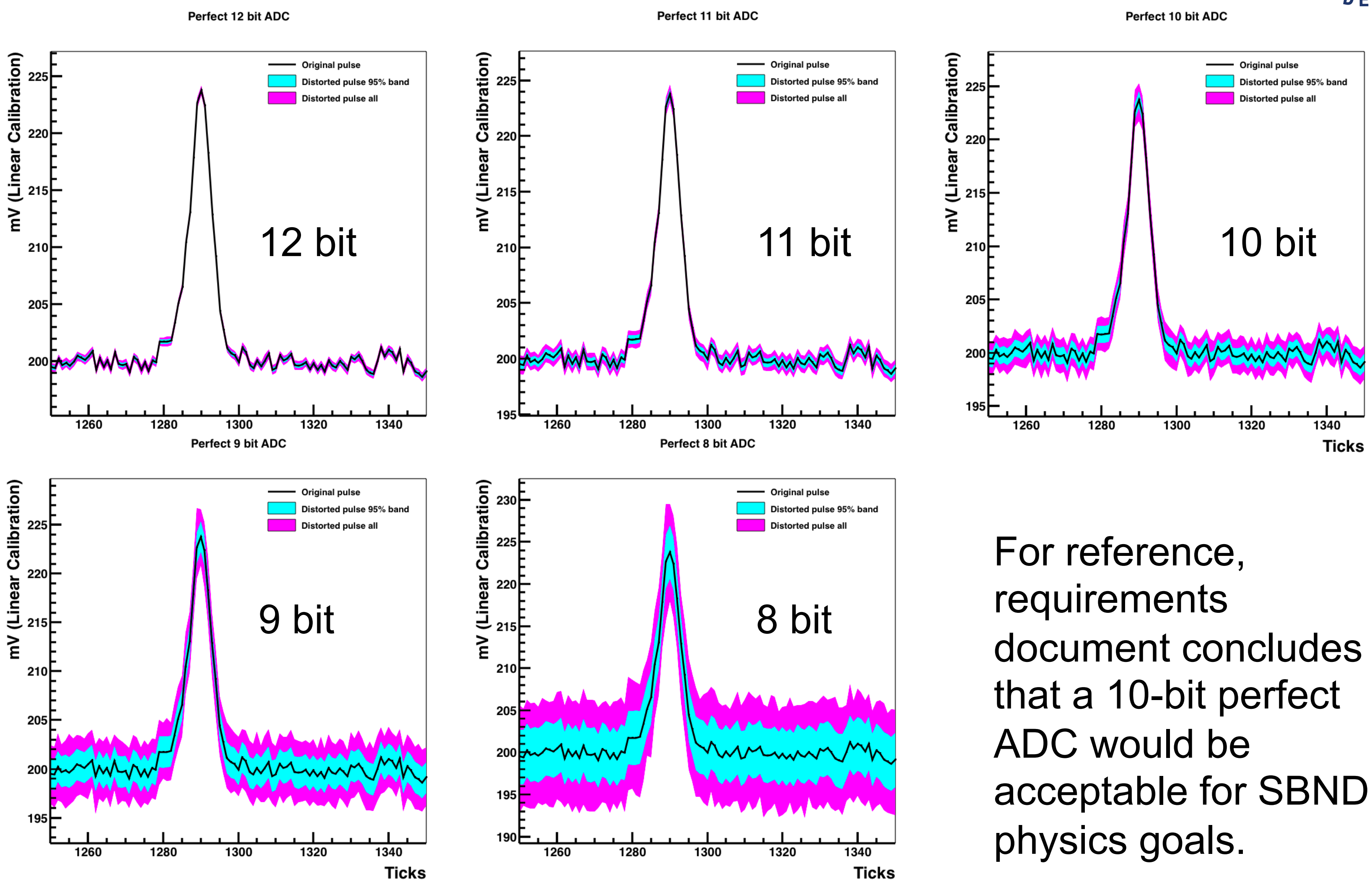




Notes on following plots

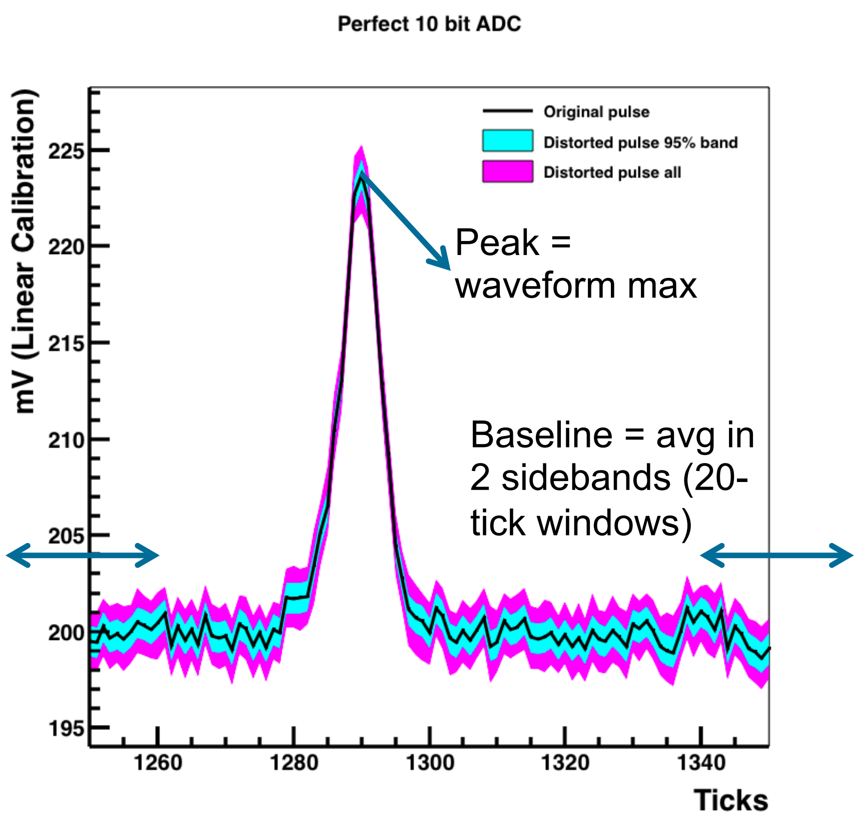
- All use the same input MIP pulse. Have expanded code to use a number of input pulses.
- 10k waveforms per plot
  - For each iteration, chip and channel are randomly selected from ~10 chips, 6 channels each. Have expanded inputs to larger sample but results not shown here. For plots with “bad” chips removed, keep sampling until a “good” chip is chosen.
  - Chips 60-69 (3 good, 4 fair, 1 poor, 2 bad according to DA proposed protoDUNE ADC classification scheme)
- No time correlations – sample randomly from ADC distribution for each tick

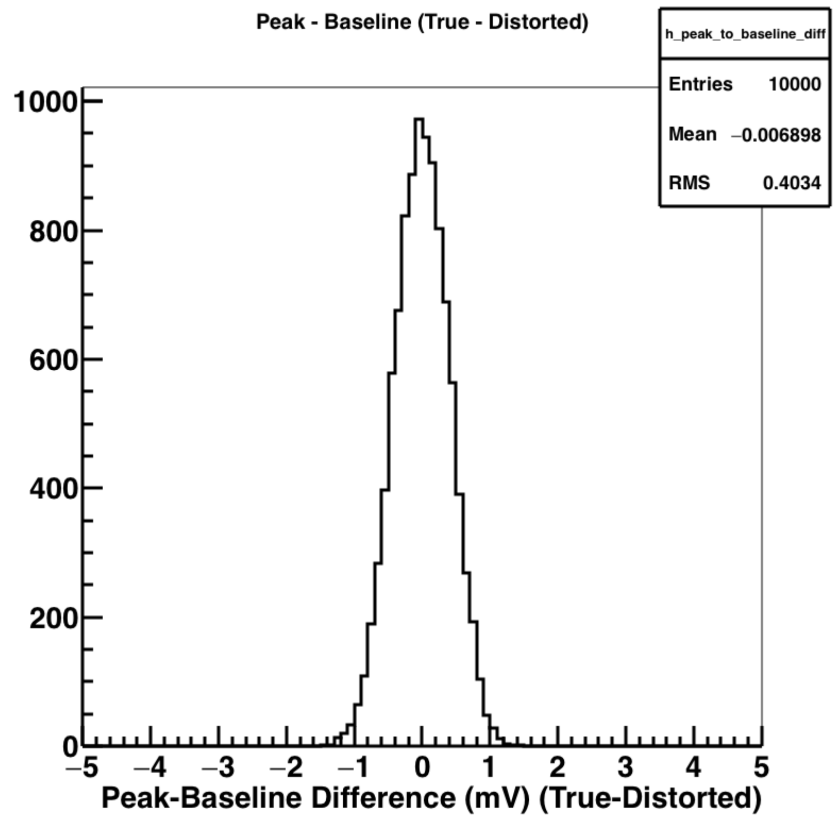
Perfect ADCs



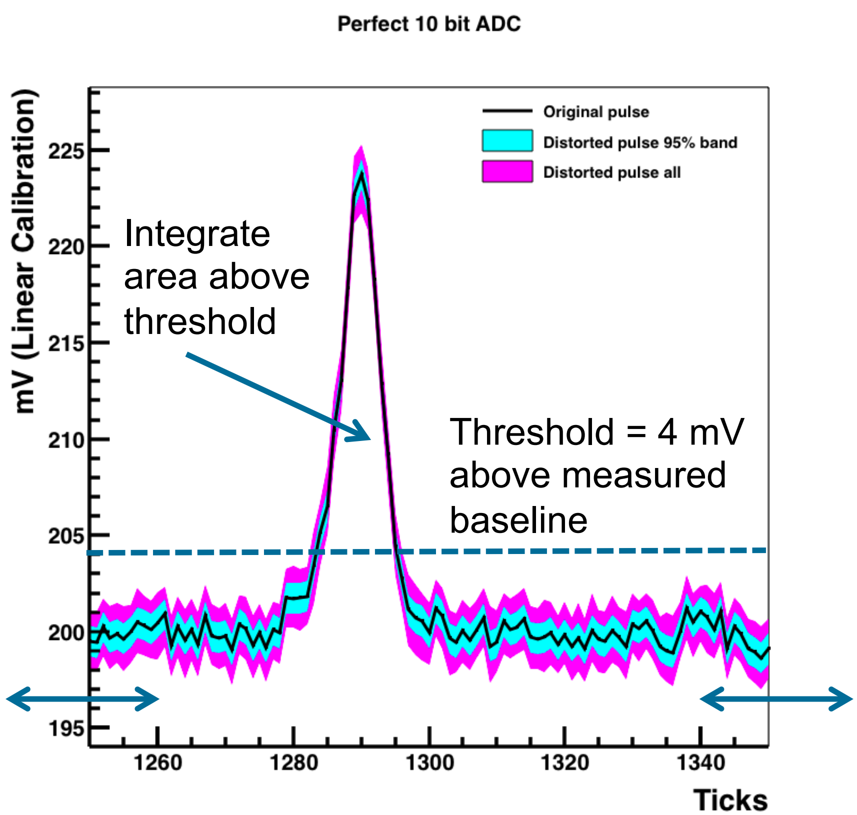
For reference, requirements document concludes that a 10-bit perfect ADC would be acceptable for SBND physics goals.

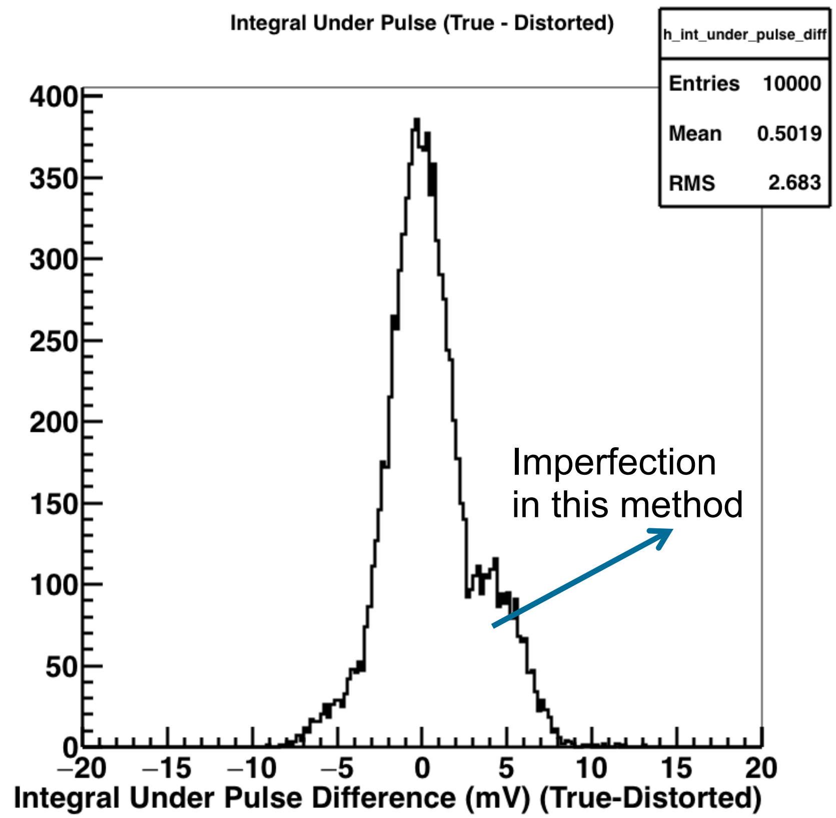
Metric: Peak to Baseline





Metric: Peak Integral





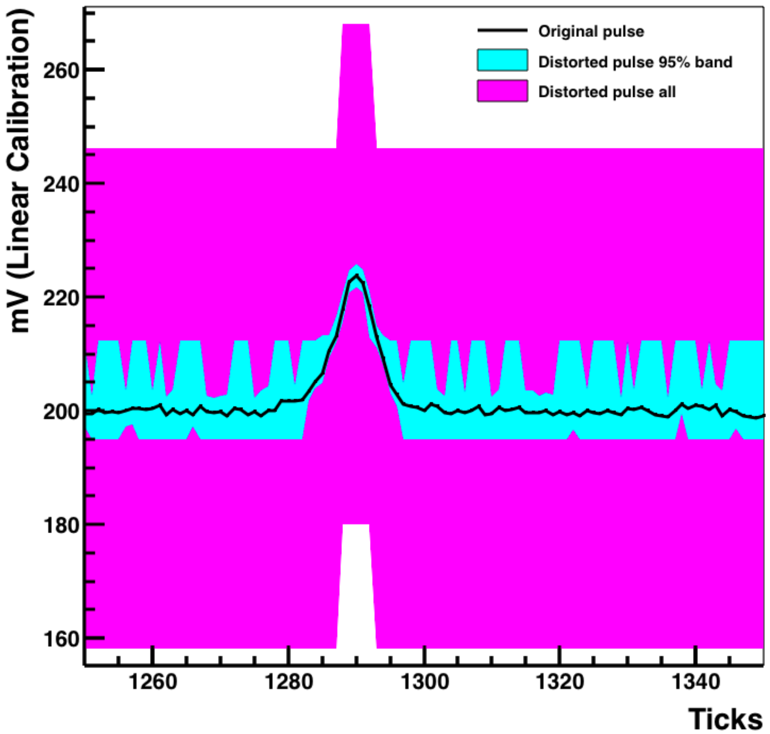
Plan to use slightly more sophisticated algorithm for this in future.

Imperfect ADCs

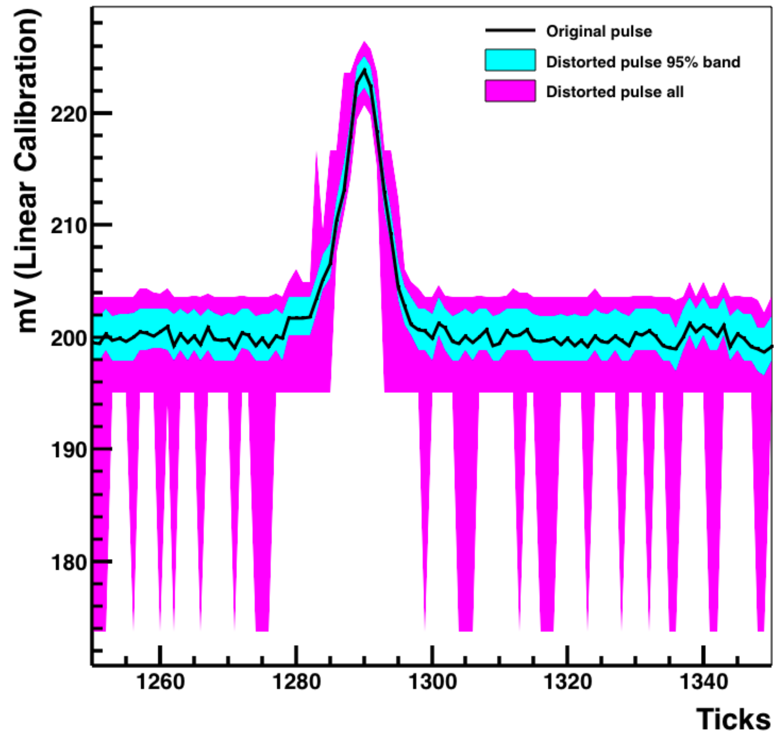
- Given the imperfections in the P1 ADC, several mitigations can be considered
  - Screening to remove poorly performing chips
  - Nonlinear calibration – here I do the “best possible” calibration, meaning one correction per code, 4096 correction constants per channel
  - Remove “classic” sticky codes (ADC%64 = 0, 63)
  - Remove measured bad codes
  - Remove measured bad codes and interpolate
  - Dual gain configuration such that one path has 4x higher gain relative to FE. Always choose higher gain if not saturated.
  - Dual gain configuration with selection of “good” code between two possible paths where both are unsaturated. (*not shown today)
- Will use metrics on previous pages to quantify impact in table at end…for now, let’s just look at the pictures.

Impact of Performance Screening

All Chips



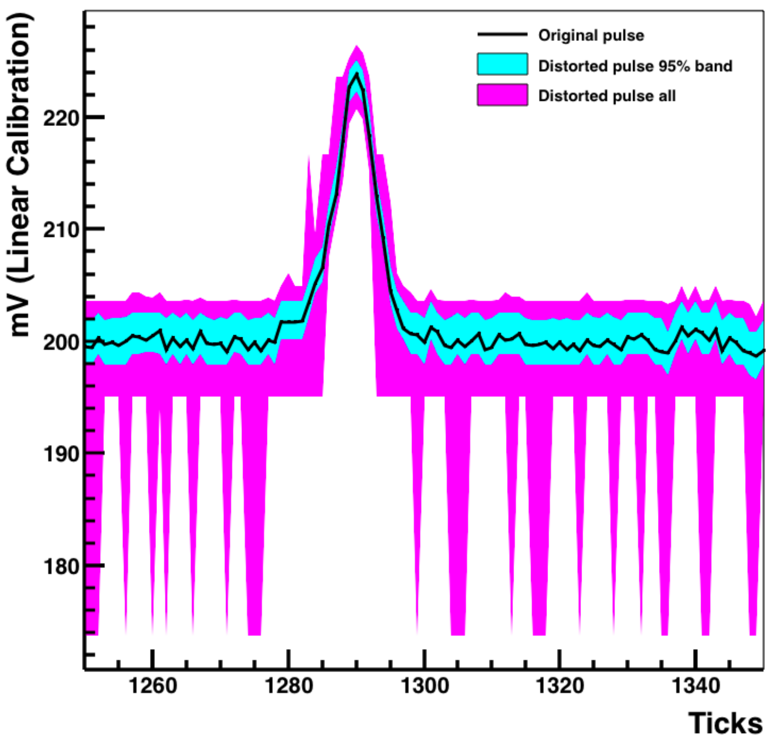
Good/Fair Chips Only



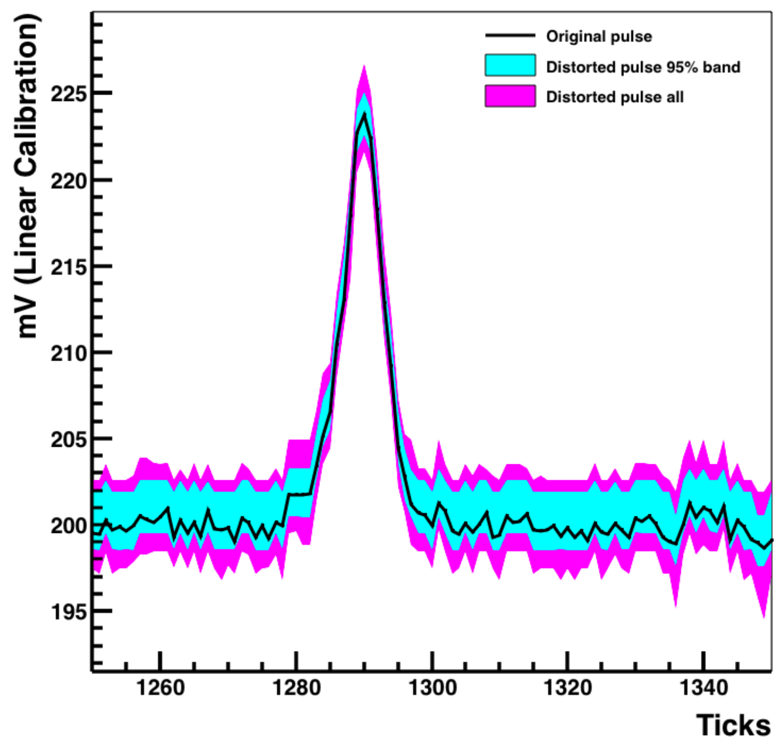
No bad code mitigation. No nonlinear calibration.

Impact of Performance Screening

Good/Fair Chips Only



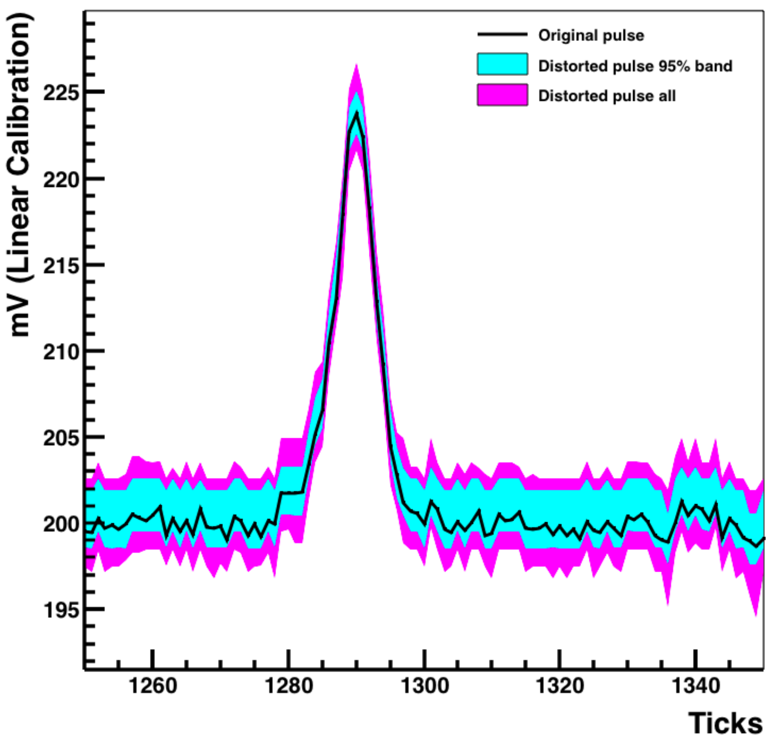
Good Chips Only



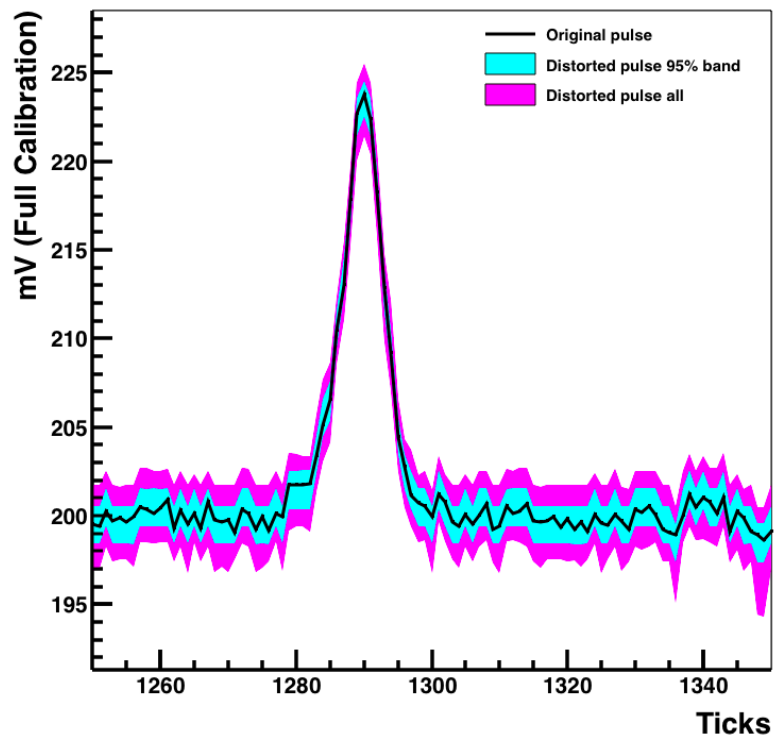
No bad code mitigation. No nonlinear calibration. (be careful, small chip stats)

Impact of Nonlinear Calibration

Linear Calibration



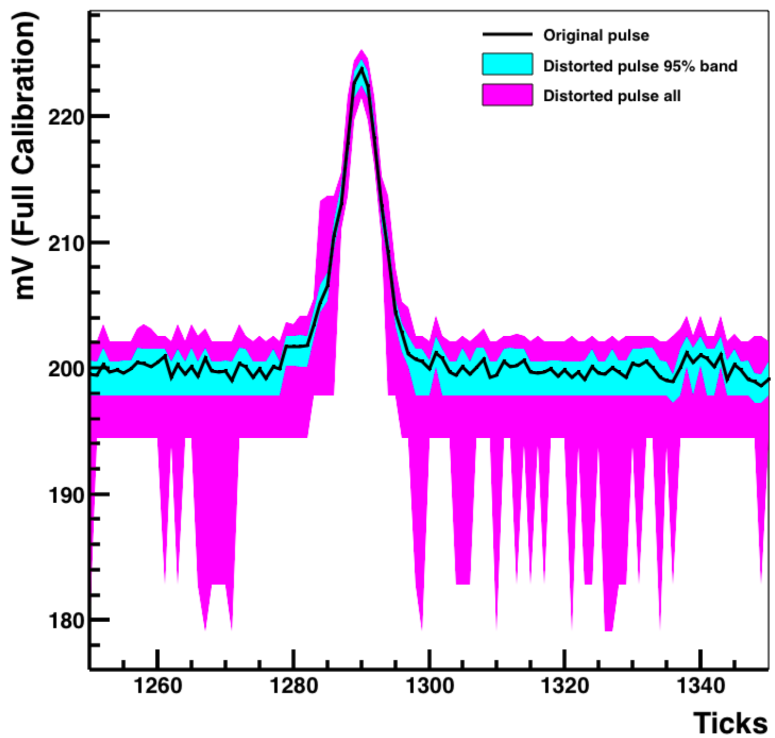
Best Calibration



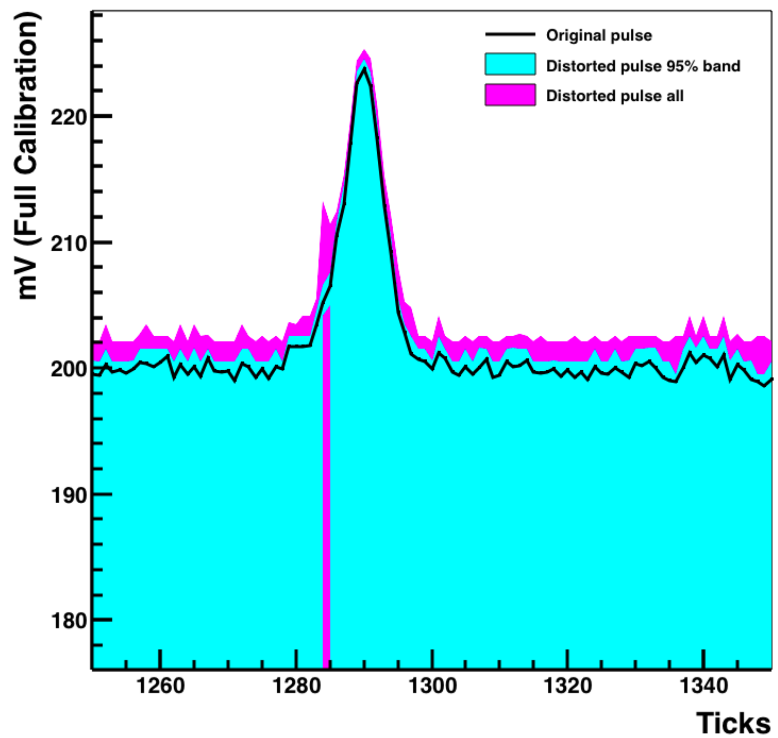
Good chips only. No bad code mitigation.

Impact of Bad Code Mitigation

No Mitigation



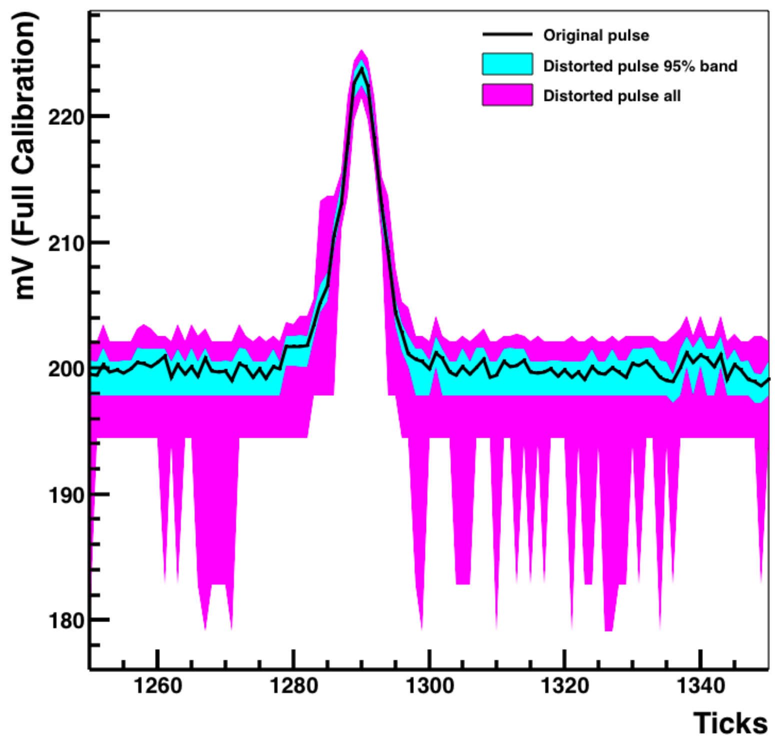
Remove “classic” sticky codes



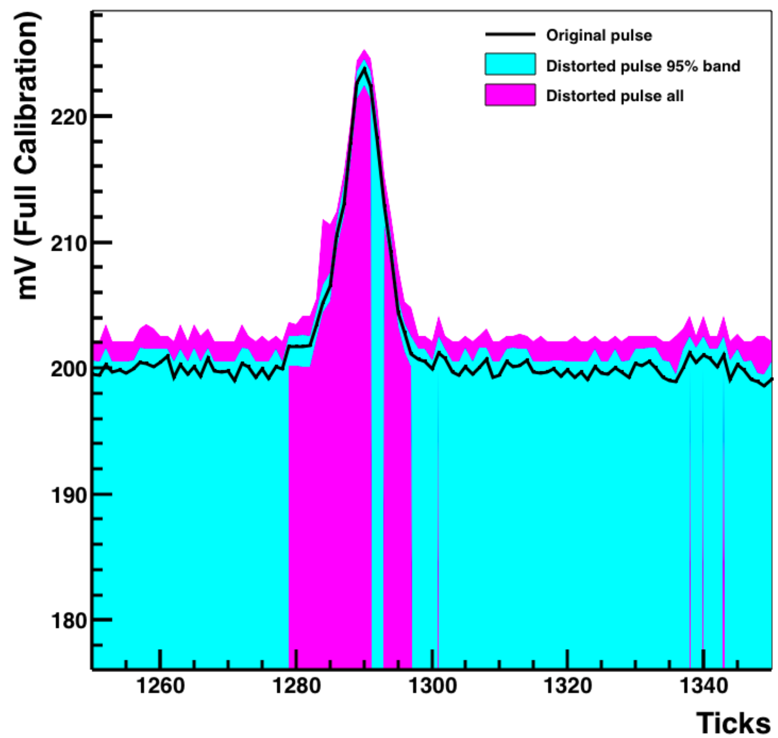
Good/fair chips only. Blind remove of classic sticky codes not great.

Impact of Bad Code Mitigation

No Mitigation



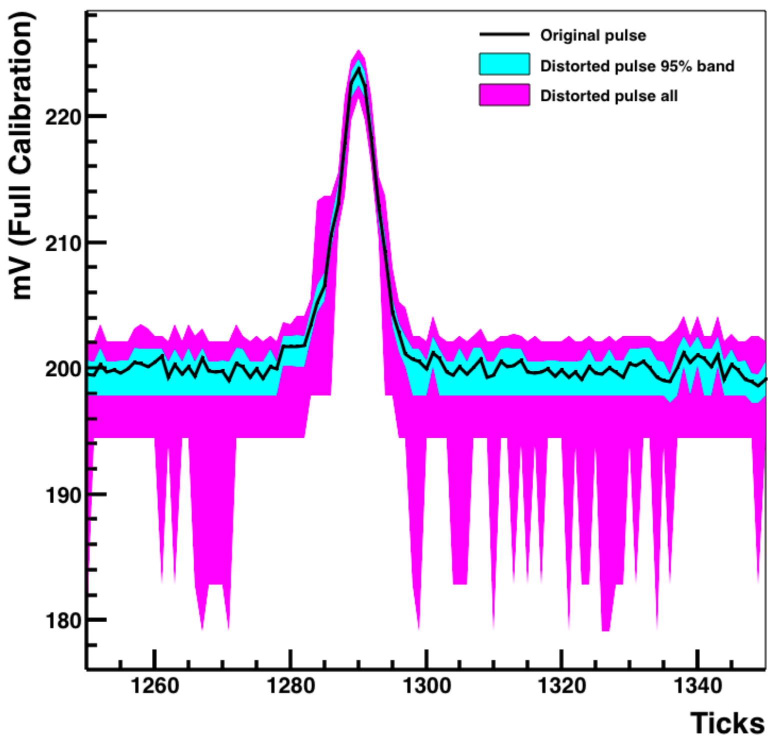
Remove bad codes



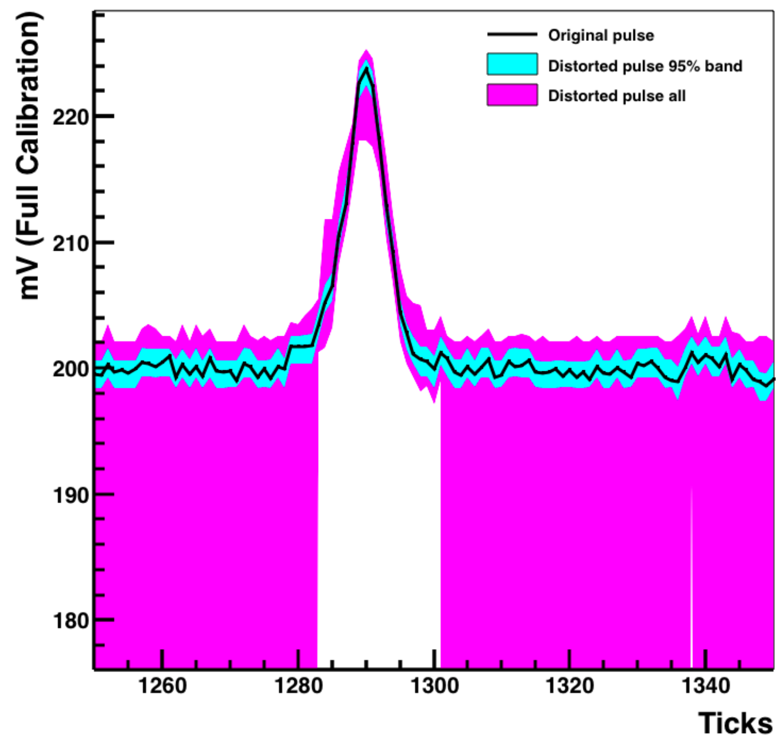
Good/fair chips only. Remove identified bad codes – still not great.

Impact of Bad Code Mitigation

No Mitigation



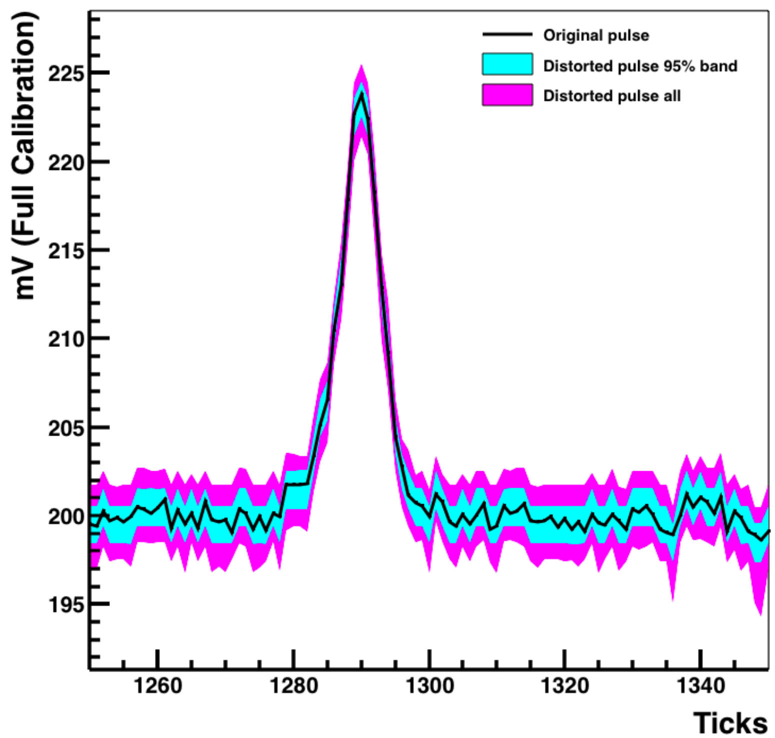
Remove bad codes and interpolate



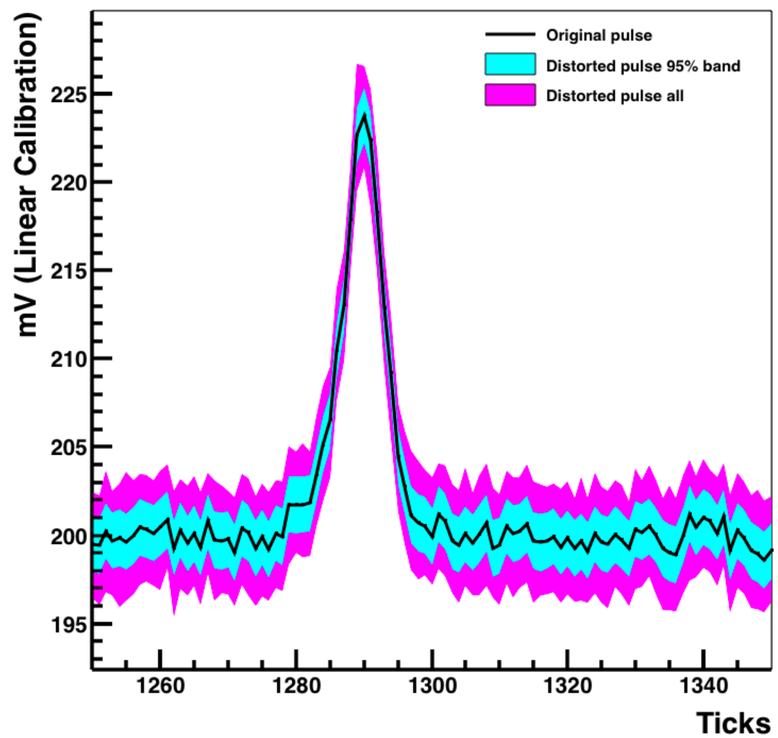
Good/fair chips only. Remove identified bad codes and interpolate up to 10 ticks away.

Best Performance of Single P1

Best Result P1



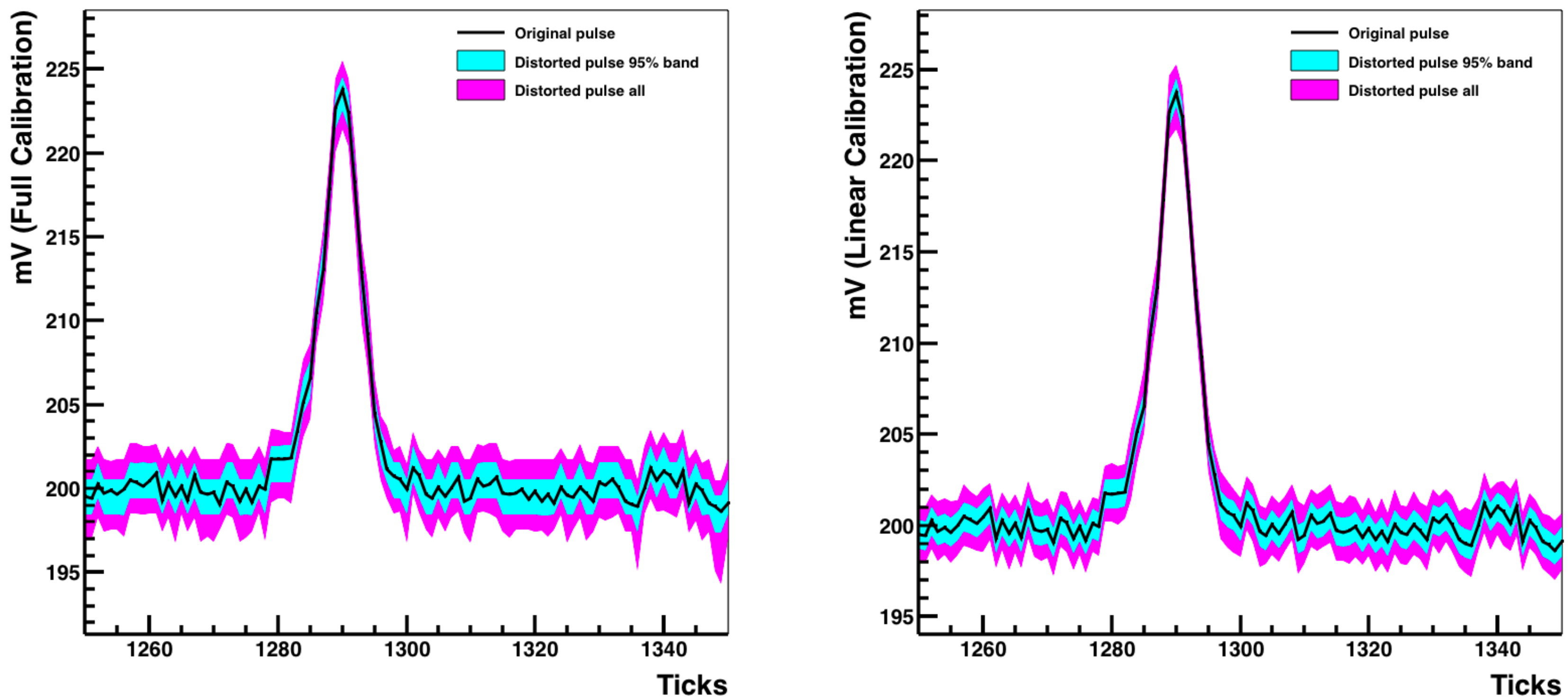
Perfect 9-bit ADC



Good chips only. Full calibration. Remove identified bad codes and interpolate up to 10 ticks away. Better than perfect 9-bit ADC.

Best Performance of Single P1

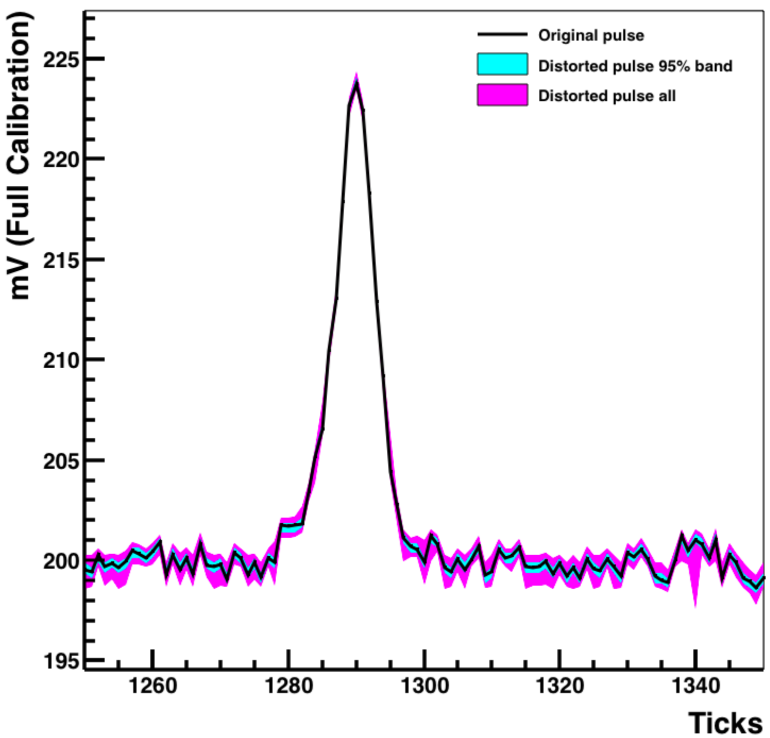
Best Result P1 | Perfect 10-bit ADC



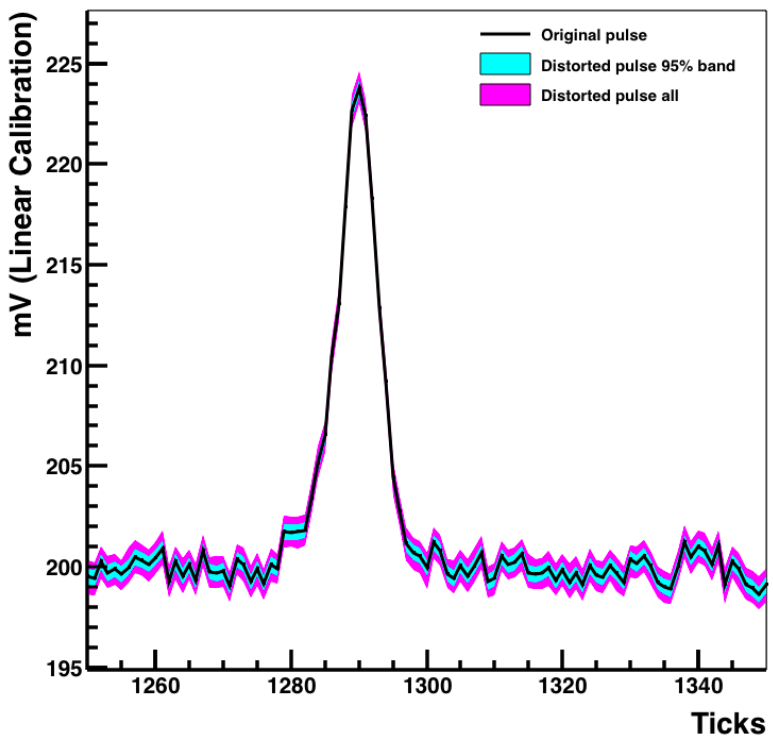
Good chips only. Full calibration. Remove identified bad codes and interpolate up to 10 ticks away. Not quite as good as perfect 10-bit ADC.

Dual Gain Option

Dual Gain



Perfect 11-bit ADC



Good/fair chips. Full calibration. Remove identified bad codes and interpolate up to 10 ticks away. Compare to perfect 11-bit ADC.

Quantify Results

Configuration	RMS Diff Peak to Baseline	RMS Diff Integral Under Peak	Compare To:
12-bit perfect	0.11	0.41	
11-bit perfect	0.21	1.21	
10-bit perfect	0.40	2.69	
9-bit perfect	0.71	4.97	
8-bit perfect	1.29	8.76	
Good/Fair, Linear Calib	0.82	5.27	8-9 bit
Good, Linear Calib	0.64	5.14	9 bit
Good/Fair, Best Calib	0.69	3.60	9-10 bit
Good, Best Calib	0.50	3.54	9-10 bit
Good/Fair, Linear Calib, Dual Gain	0.22	1.38	Nearly 11 bit
Good/Fair, Best Calib, Dual Gain	0.12	0.62	Nearly 12 bit

(Preliminary) Conclusions

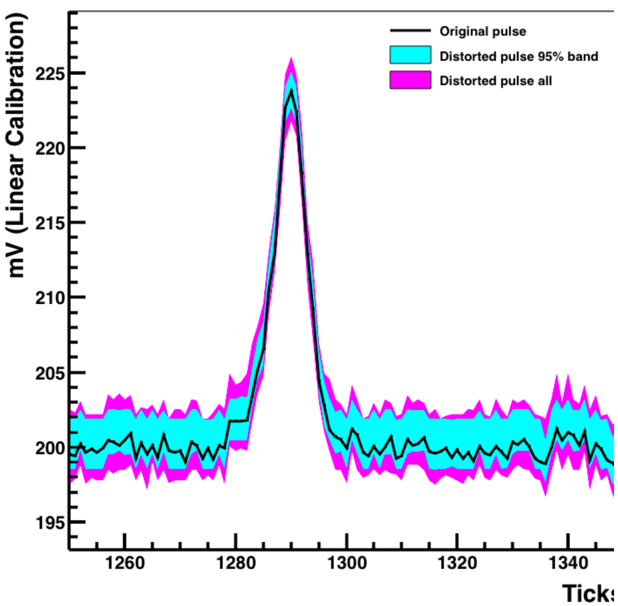
- Quick-and-dirty waveform simulation tool exists – easy for me to make changes and run additional studies
- Simulation results confirm back-of-the-envelope calculations
  - Best result for P1 alone is marginal for SBND requirements
  - Results for x4 dual gain meet SBND noise/resolution requirements
- Non-linear calibration important for good performance
- If chips are screened, bad codes are not a big issue for performance (affect a few percent of data and can be mitigated)
- To do:
  - More test stand data (more chips/channels) in simulation
  - More waveforms in simulation
  - Check induction wire baseline
  - Use dual-gain for bad code mitigation
  - Investigate less-aggressive interpolation
  - Investigate less finely-binned non-linear calibration

Updates

- Cleaned up code a bit, put in github
- Include more chips/channels ✓
  - My simple mod to David's code running now – almost done
- Include more waveforms in simulation
  - More of these parallel MIP pulses ✓
  - Re-evaluate with new pulses from Michelle
  - Re-evaluate with new pulses from Mike/Adam standalone sim (including larger pulses)
- Check other baselines ✓
  - Artificially set baseline at range of values – no large impact ✓
  - For dual gain, look at pulses on threshold between gains
- Use chip yield fraction rather than arbitrary "good/fair/poor" definitions
  - Got ordered list from David's results, not yet implemented
- Implement 64-bin calibration ✓

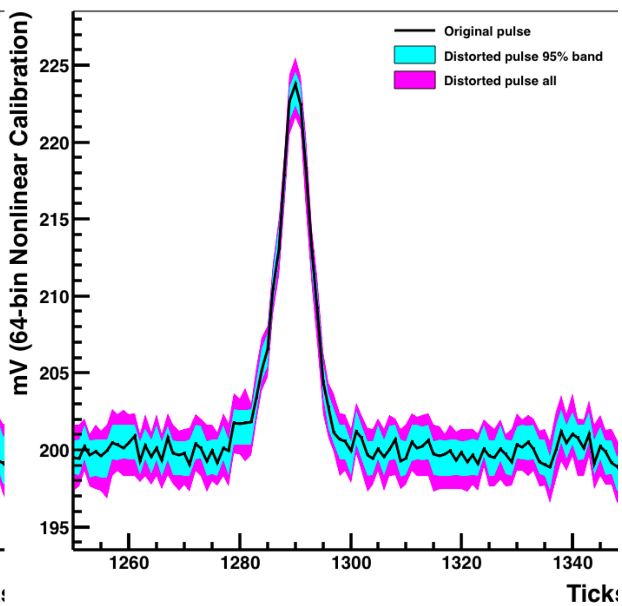
First Look at 64-bin Calib

Linear



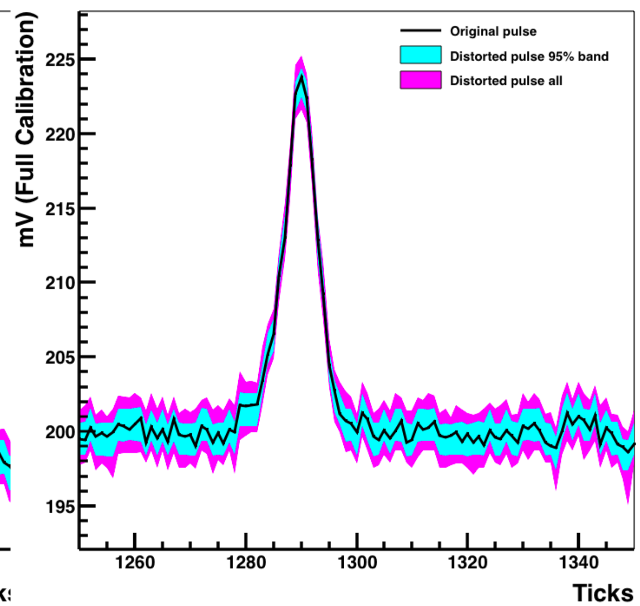
PtB Diff RMS = 0.65 mV

64 bin



PtB Diff RMS = 0.55 mV

4096 bin



PtB Diff RMS = 0.48 mV

*Only 1000 samples here, single waveform